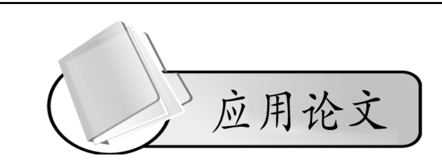


DROID 开源工具在长期保存系统格式识别中的应用

王玉菊　吴振新　孔贝贝　付鸿鹄
(中国科学院文献情报中心　北京 100190)

摘要: 【目的】在数字资源长期保存系统(DPS)中应用开源格式识别工具获取复杂对象的格式信息。【应用背景】在现有开源工具的基础上, 为满足 DPS 的实际需求、保障效率和执行效果, 需选择合适的工具进行二次开发和集成应用。【方法】分析比较现有两种常用工具, 选取 DROID 作为 DPS 的格式识别工具, 同时针对 DPS 效率要求, 提出选用 DROID 批量格式识别的处理思路, 并对其进行有效封装。【结果】将 DROID 封装为"DPS 的批量格式处理模块"并在 DPS 格式识别及技术元数据抽取中得到实际应用。【结论】 DROID 是一个优秀的开源工具, 其自动批处理特性基本满足 DPS 格式处理需求。

关键词: 格式识别　长期保存　复杂对象
分类号: TP393

网络时代, 知识和数据载体已从传统的印本形式向数字化形式转变, 而每种数字化的资源都有其独特的格式, 英国 DCC(Digital Curation Center)在其《数字保存指南》[1](Digital Curation Manual)中, 专门针对文件格式问题指出, "格式是数字对象的基本特征, 它在很大程度上决定了数字对象的有效性, 没有适当的格式, 无法识别数字对象的内容, 数字对象就是一组无意义的比特流。"随着时间的推进, 数字对象的格式不可避免地会过时, 为维护数字对象的长期可生存能力、可呈现能力和可理解能力, 格式迁移作为一种有效的保存策略, 被众多项目所采用, 并根据不同的实践需求实施了不同的格式管理措施。技术元数据作为长期保存元数据的一部分, 既是格式管理的核心内容, 又是数字对象格式迁移的基础。而在目前的数字资源长期保存活动中, 要保存的数字对象已不再是单纯的 PDF 文件, 涉及到音频、视频、图片、文本等多种文件格式(本文统称这些对象为复杂对象), 如何正确有效获取复杂对象的格式、版本等信息, 并把这些信息作为技术元数据进行长期保存和管理也成为长期保存系统需要解决的关键问题。

国内外提供多种数字对象格式识别工具, 用来获取不同格式数字对象的基本信息、格式信息、可依赖环境信息等, 很多机构也根据实际需求选择合适的格式识别工具进行格式管理。中国科学院文献情报中心数字资源长期保存系统(DPS)在研发过程中也制定了相关的格式管理策略, 利用现有格式识别工具对所有的文档进行格式注册管理并定期进行格式跟踪和监测。本文就是基于 DPS 的这些实际需求, 重点探讨了如何选择符合要求的格式识别工具, 并通过二次开发封装以更好地保障保存系统的效率和执行效果。

1　常用格式识别的主要工具及其选型

当前在长期保存系统中常用的格式识别工具有 JHOVE 和 DROID 两种。简单对二者进行介绍, 并从

通讯作者: 王玉菊, ORCID: 0000-0003-2539-2218, E-mail: wangyj@mail.las.ac.cn。

应用环境、使用方式、识别文件类型、识别信息、结果集呈现格式、可识别文件类型的扩展性、批量处理的效率等对二者进行对比分析，然后从抽取技术元数据信息和可识别文件类型的角度考虑，进行工具选型。

1.1　主要工具

JHOVE[2] (JSTOR/Harvard Object Validatio Environment) 是美国哈佛大学和 JSTOR 共同研制的数字对象管理工具，主要用于数字对象的格式标识、验证和特征描述。其中，格式标识就是确定一个数字对象的格式，回答“有一个数字对象，那么是什么格式?”这问题；格式验证就是确定一个数字对象与其他的格式描述的吻合程度，回答“有一个数字对象，它的格式是 F，是吗?”这一问题；格式特征描述是确定一个给定格式的数字对象的重要特征，回答“有一个数字对象，它的格式是 F，那么它的显著特征是什么?”这一问题。JHOVE 是基于 Java 的开源工具，通过 12 个标准模块支持文件格式的标识、验证和特征描述。当前 JHOVE 的最新版本为 1.6，它提供图形界面和命令行两种使用方式，可选择对单个文件处理，也可对文件夹进行递归处理。目前，DATISS2[3]应用 JHOVE 进行文件格式验证并提取技术元数据、Portico[4]应用 JHOVE 提取验证状态和技术元数据、UAM [5]应用 JHOVE 进行文件格式识别并提取技术元数据、DIAS[6]应用 JHOVE 抽取技术元数据。

DROID[7](Digital Record Object IDentification)是 2005 年，由英国国家档案馆数字资源长期保存小组为了识别并存储其长期保存数字对象格式而开发的数字对象自动识别工具。DROID 也是基于 Java 语言的开源工具，使用数字对象的内部和外部特征来识别数字对象的文件格式、版本及依赖环境等信息，这些数字对象特征信息都存储在一个 XML 格式的“特征文档”(Signature File)中，“特征文档”由 PRONOM(在线技术注册中心)定期更新维护，DROID 通过互联网方式实时地到 PRONOM 上抓取最新的“特征文档”用作格式识别，当前提供的最新“特征文档”版本为 V77。DROID 的最新版本为 6.1.3，需与互联网连接，以便于“特征文档”的自动更新。此工具操作简单，可选择文件或文件夹进行识别，只需几步即可完成对文件格式的识别，识别速度快、可批量实现对数字文档的识别，目前在 DATISS2[3]、KB Preservation Manager、Planets[8]、PRESERV[9]等项目应用 DROID 进行格式识别。

1.2　工具选型

主要从 JHOVE 和 DROID 的应用环境、使用方式、识别文件类型、识别信息、结果呈现格式等方面对二者进行对比分析，如表 1 所示:

表 1　JHOVE 和 DROID 主要特征对比

比较项	JHOVE	DROID
应用环境	跨平台，JRE1.6.0 及其以上，可应用在 Unix，Windows 或 OS X 平台上	跨平台，JRE1.6.0 以上，最小 512MB 内存，需与互联网连接，可应用在 Windows2000、XP、Vista、Macintosh OS X、Red Hat Enterprise Linux、SUSE、Debian、Ubuntu 平台上
可识别文件类型	PDF、TXT、GIF、JPEG、JPEG2000、TIFF、AIFF、WAV、HTML、XHTML、XML 等	DBF、DOC、Lotus Formats、MS Works Formats、OpenOffice Formats、MDB、MPP、PDF、PPT、PST、PUB、RTF、StarOffice Formats、TXT、VSD、WPD、WS and Other WordStar Formats、XLS、BIFF BMP、CDR、Corel Formats、DWG、AutoCad Formats、DXF、EPS、GeoTIFF、GIF、JPEG、JPEG2000、PageMaker Documents、PCX、PNG、PS、PSD、PSP、SWF、Macromedia Formats、SVG、TIFF、AIFF、ASF、AVI、MIDI、MOV、MP3、MPG、Real Audio (RM/A)、WAV、GML、HTML、ODF、XML、XHTML、JS、TAR (Tape Archive Format)、ZIP 等
识别信息	路径或 URI、最近修改日期、大小、格式、格式版本、MIME Type、格式配置文件、校验码	路径或 URI、最近修改日期、类型、大小、名称、扩展名、扩展名是否匹配、格式识别数量、格式名称、格式版本、PRONOM 唯一标识符、MIME Type、识别方法、MD5 码、文件状态
结果文件类型	TEXT, XML	XML, CSV

从表 1 中分析可见，二者都是跨平台的，都能识别文本、PDF、音视频、图片等文件类型，都能提供 XML 识别结果集。JHOVE 除了提供格式识别外，还具有格式验证、格式特征描述的功能，能获取文件相关的软件信息，其抽取的特征属性也更丰富；对于 JHOVE 可以识别的格式类型，JHOVE 和 DROID 在性能表现方面基本相当，但对于 JHOVE 无法验证的格式类型，JHOVE 与 DROID 相比识别性能会有所下降。

相对 JHOVE 而言，DROID 识别的文件类型更多，DROID 与 PRONOM[10]技术信息注册中心联盟，对当前无法识别的文件格式，支持用户在线自助注册要扩展的文件类型，PRONOM 将新增格式定期更新到“特征文档”中，同时还可以通过 PRONOM 唯一标识符到 PRONOM 上获取文件的生产软件、生产商、文件生命周期、迁移方法等信息。另外，DROID 本身是因长期保存数字对象存储而开发的，其提供的自动批量格式识别更接近笔者的需求，因此从多角度综合考虑，DROID 工具是一种更为合适的选择。

2 DROID 主要功能原理及其识别实例

2.1 主要功能原理

DROID 提供 Extension、Signature 和 Container 三种识别方法。

(1) Extension[11]识别是完全通过文件扩展名来识别文件格式，因为文件可以选择任何方式来命名，所以这种识别方法不是很可靠，同时 Extension 识别也无法识别到格式的版本级别，有时同一文件还会识别出多种格式结果，因此该识别方式比较粗泛。

(2) Signature[11]识别是针对一类特定数字对象，这类数字对象的内部包含格式的特征属性，Signature 识别方法通过数字对象的内部特征来识别数字对象格式及版本，因为数字对象的内部特征是唯一的、且不易被外界改变，因此这种识别模式非常有效可靠。

(3) Container[11]识别是通过识别内嵌数字对象来获取文件格式和版本信息，以 OpenDocument 为例，一个 OpenDocument 是一个包含多个 XML、Images 等数字对象的 ZIP 文件，Container 识别方法将识别该数字对象的格式是 OpenDocument 而不是 ZIP。Container 识别方式不仅对容器进行识别(比如 ZIP)，它还将压缩文件打开，对压缩包内的数字对象进一步识别，这种识别方法底层也是使用 Signature 方式，因此 Container 也是一种非常可信的识别模式。

对于上述三种识别方式，DROID 优先选用 Signature，如果 Signature 识别未获取到数字对象格式，则依次选用 Container、Extension 识别方法来完成文件格式的识别。

DROID 既提供对单个文件的格式识别，也提供对一个目录的格式识别，即 DROID 是以一个文件或目录作为输入，识别出文件的路径或 URI、最近修改日期、类型、大小、名称、扩展名、扩展名是否匹配、格式识别数量、格式名称、格式版本、PRONOM 唯一标识符、MIME Type、识别方法、MD5 码、文件状态信息，最后以 XML、CSV、Printer-Friendly Formats 格式提供给第三方使用。

DROID 提供了 GUI 和命令行两种使用方式，能识别 1 000 多种文件格式，对无法识别的文件格式，DROID 通过 PRONOM 提供用户在线自助扩展功能。下面以命令行方式为例，分析 DROID 工具的整个识别过程。

2.2 识别过程分析

DROID 工具提供的 API 都是命令行方式，它是一种伪 API，不是真正意义上的 API 接口，本文中简称 DROID-API，该接口可单独使用，也可简单与其他系统集成。DROID-API 接口不允许所有的操作集中在一个单独的命令行完成，需要分两个阶段完成整个文件的格式识别。图 1 是使用 DROID-API 接口对一个文件或文件夹进行识别的全流程。

(1) 文件格式识别

第一阶段是创建新的 Profiles 并将格式识别结果保存到 DROID 的内部文件。

DROID 首先为即将识别的文件或文件夹自动创建一个空白 Profiles，一个 Profiles 是一个单独的目录，该目录下包含最新版本的 SignatureFile、数据库、索引、日志、Profile.xml 文件，Profiles 主要用来管理并记录格式识别过程使用的 SignatureFile、识别过程和识别结果。图 2 是一个 Profiles 的目录结构。

其中，Profile.xml 文件内容记录了 Profiles 的唯一标识、创建日期、使用的 SignatureFile 文件及其版本、是否启用 MD5 等，详细内容如图 3 所示。

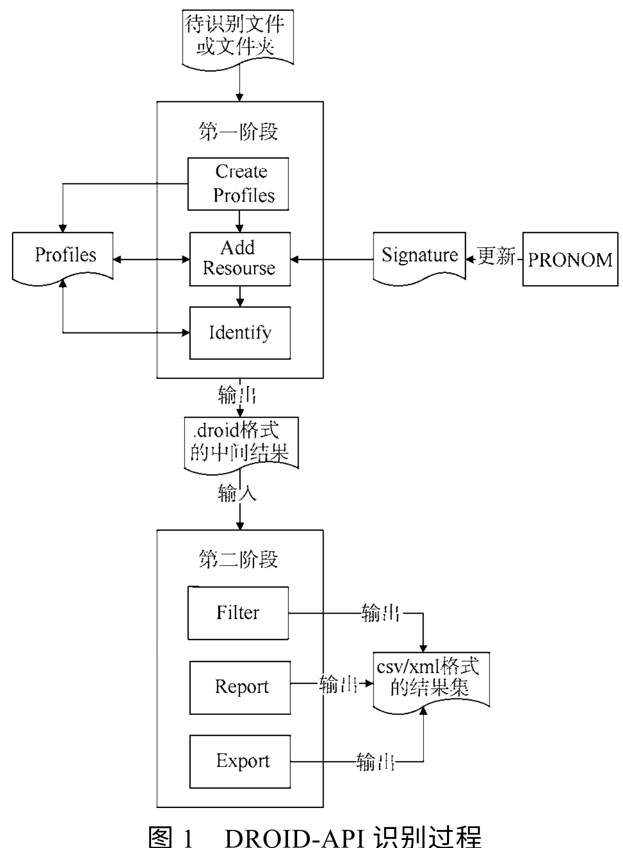


图 1　DROID-API 识别过程

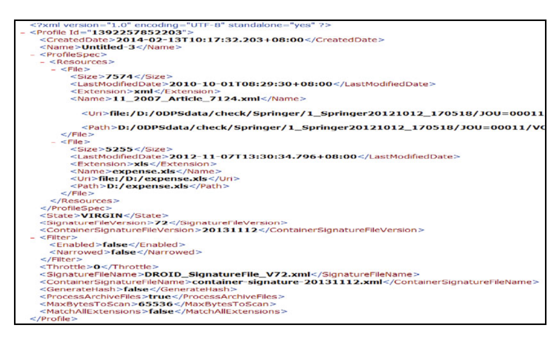
```
db                                    log
container-signature-20131112.xml      seg0
DROID_SignatureFile_V72.xml           tmp
profile.xml                           db.lck
                                      service.properties
```

图 2　Profiles 的目录结构

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
- <Profile Id="1392257852203">
    <CreatedDate>2014-02-13T10:17:32.203+08:00</CreatedDate>
  - <ProfileSpec>
      <Resources />
    </ProfileSpec>
    <SignatureFileVersion>72</SignatureFileVersion>
    <ContainerSignatureFileVersion>20131112</ContainerSignatureFileVersion>
    <Throttle>0</Throttle>
    <SignatureFileName>DROID_SignatureFile_V72.xml</SignatureFileName>
    <ContainerSignatureFileName>container-signature-20131112.xml</ContainerSignatureFileName>
    <GenerateHash>false</GenerateHash>
    <ProcessArchiveFiles>true</ProcessArchiveFiles>
    <MaxBytesToScan>65536</MaxBytesToScan>
    <MatchAllExtensions>false</MatchAllExtensions>
</Profile>
```

图 3　Profile.xml 的详细内容

Profiles 创建完成后，将要识别的文件或文件夹加载到上述刚刚创建的 Profile.xml 中，在加载过程中获取文件名、扩展名、路径、URI、大小、最新更新日期追加到 Profile.xml 文件中，以文件 11_2007_Article_7124.xml 和 expense.xls 为例来展示 DROID 工具的“Add resource”的过程，如图 4 所示：



```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
- <Profile Id="1392257852203">
    <CreatedDate>2014-02-13T10:17:32.203+08:00</CreatedDate>
    <Name>Untitled-3</Name>
  - <ProfileSpec>
    - <Resources>
      - <File>
          <Size>7574</Size>
          <LastModifiedDate>2010-10-01T08:29:30+08:00</LastModifiedDate>
          <Extension>xml</Extension>
          <Name>11_2007_Article_7124.xml</Name>
            <Uri>file:/D:/ODPSdata/check/Springer/1_Springer20121012_170518/JOU=00011
            <Path>D:/ODPSdata/check/Springer/1_Springer20121012_170518/JOU=00011/VC
        </File>
      - <File>
          <Size>5255</Size>
          <LastModifiedDate>2012-11-07T13:30:34.796+08:00</LastModifiedDate>
          <Extension>xls</Extension>
          <Name>expense.xls</Name>
          <Uri>file:/D:/expense.xls</Uri>
          <Path>D:/expense.xls</Path>
        </File>
      </Resources>
    </ProfileSpec>
    <State>VIRGIN</State>
    <SignatureFileVersion>72</SignatureFileVersion>
    <ContainerSignatureFileVersion>20131112</ContainerSignatureFileVersion>
  - <Filter>
      <Enabled>false</Enabled>
      <Narrowed>false</Narrowed>
    </Filter>
    <Throttle>0</Throttle>
    <SignatureFileName>DROID_SignatureFile_V72.xml</SignatureFileName>
    <ContainerSignatureFileName>container-signature-20131112.xml</ContainerSignatureFileName>
    <GenerateHash>false</GenerateHash>
    <ProcessArchiveFiles>true</ProcessArchiveFiles>
    <MaxBytesToScan>65536</MaxBytesToScan>
    <MatchAllExtensions>false</MatchAllExtensions>
</Profile>
```

图 4　加载待识别源后的 Profile.xml 内容

DROID 对已经加入到 Profile.xml 中的文件或文件夹进行格式识别，在此，要注意一旦开始对文件进行识别，不能再向刚刚创建的 Profiles 中再添加任何文件和目录，如果还想识别其他的文件，需要重新创建一个 Profiles。

如上所述，DROID 完成了指定文件或文件夹的格式识别，并将识别结果保存到.droid 格式的内部文件，完成第一阶段的识别工作。

(2) 结果输出

第二阶段是将识别结果通过 Filter、Report、Export 的形式转为 CSV 或 XML 格式的外部结果集文件供第三方使用。

以 Export 方式为例，即使用下述命令：“droid –p.droid 文件路径 –e.csv 文件绝对路径”，将识别结果保存到 CSV 文件中，得到样例结果如图 5 所示，到此，完成整个文件的格式识别。

第三方如想使用 DROID 识别出的结果集，需要对 CSV 或 XML 文件进行解析获取需要的格式信息即可。

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
ID	PARENT_ID	URI	FILE_PATH	NAME	METHOD	STATUS	SIZE	TYPE	EXT	LAST_MODI	EXTENSION	MD5_HASH	FORMAT_C	PUID	MIME_TYPE	FORMAT_
1		file:/D:/	D:\ODPSda	11_2007_A	Signature	Done	7574	File	xml	2010-10-(	FALSE		1	fmt/101	applicati	Extensi
2		file:/D:/	D:\expens	expense.x	Signature	Done	5255	File	xls	2012-11-(	TRUE		1	fmt/100	text/html	Hyperte

图 5　DROID 的识别结果

3　DROID 在长期保存系统中的集成应用

中国科学院文献情报中心数字资源长期保存系统在数据格式接收中有自己的政策要求。

(1) 要求数据资源提供商按业界公共的或通用的标准格式提供元数据、全文数据以及附加资料数据，以便建立公共的格式解析、内容检验和数据转换程序；

(2) 除非因特殊原因无法提供公共格式数据、且得到中国科学院文献情报中心同意后，才允许数据资源提供商提供不同于公共格式的数据格式，并且不允许对方提供完全私密的格式。

基于以上政策要求，中国科学院文献情报中心数字资源长期保存系统(DPS)目前接收存档数据特点是：数据量大，同时涉及到 PDF、图片、音视频、表格、文本等多种格式的数字对象。而 DPS 是以每次接收的 SIP 集合为处理对象，每个 SIP 集合少则包含上千、多则包含上百万、上千万个数字对象，DPS 需要对每个数字对象进行格式识别并抽取技术元数据进行长期保存。

如果在 DPS 中直接使用 DROID-API 接口，根据现有存档流程，需要为每个数字对象生成一个格式为.droid 的中间文件，再将该文件转换成格式为 CSV 或 XML 的最终结果集文件，最后对结果集文件实现解析获取需要的技术信息进行存档，这种使用方式会反复创建、读写中间文件，从减少对中间文件的读写操作、提高 DROID 识别效率上考虑，笔者调整了 DPS 的存档流程，并对 DROID 源码进行二次封装和集成，即在存档流程之前增加“DROID 批量格式识别”过程，然后将其结果集应用在 DPS 中，具体流程如图 6 所示：

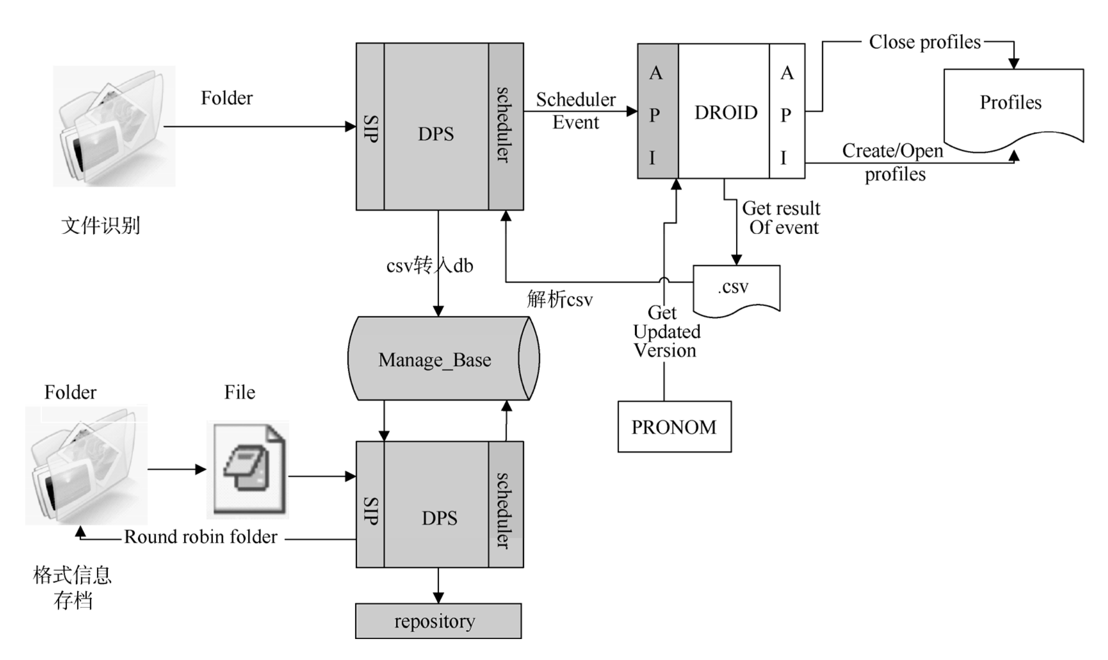


图 6　DROID 在 DPS 应用

(1) 使用 DROID 完成批量格式识别

调整了 DPS 的存档流程，即存档流程之前增加“批量格式识别”处理过程，将 DROID 第一阶段和第二阶段的 DROID-API 进行封装，使其一次完成批量格式识别和结果输出。例如：一批数据有 4 万篇文章、每篇文章有 1 个附件，即将上述 8 万个文件一次性进行批量格式识别，生成可供第三方使用的.csv 结果集文件。即调用一次 DROID-API，创建 1 个 Profiles、打开、关闭一次 Profiles，一次将内部结果.droid 转换为.csv 结果集，批量识别方式减少了文件的创建、打开、关闭操作，提高了处理效率，减小了系统开销。

(2) 识别结果集在 DPS 中的应用

针对上述生成的.csv 格式的结果集文件，如在 DPS 中直接使用，以上述数据为例，需要进行 8 万次打开、读取文件操作，必然增加系统开销。因此，考虑将.csv 文件内容导入数据库临时表，在存档过程中以文件的“FILE_PATH”为检索条件，从临时表中唯一获取文件格式信息进行长期保存。由于数据库读写技术

相对成熟完善，导入临时表后，可为需要检索的关键字段建立索引，提高检索效率，相比较对.csv 文件的操作，可大大提高检索速度及准确度。

4 DROID 的评价指标分析及验证

笔者结合数字资源长期保存项目实际需求，在使用DROID进行格式识别实验时，着重从以下6个评价指标对 DROID 进行分析和验证。

(1) 识全率

格式识别工具能识别的文件类型是目前项目比较关注的一个评价指标。

DROID 使用文件内部和外部特征来识别文件格式，这些数字对象特征信息都存储在一个 XML 格式的"特征文档"中，"特征文档"由英国国家档案馆定期进行更新，目前 V77 的"特征文档"中包含 1 400 多种文件格式。同时实验表明，目前 DPS 涉及到的 99%以上的文件格式都出现在"特征文档"中。

(2) 可扩展性

DROID 在新增文件格式和对已有文件格式特性的更新上具有可扩展性。

DROID 能识别的文件格式都记录在"特征文档"中，可以通过新增或更新"特征文档"进行文件格式的新增和更新。

(3) 识别粒度

DROID 格式识别的粒度到 PUID(PRONOM Unique IDentifiers)和 MIME Type，它可以通过 PUID 直接关联到"在线技术注册中心"。

(4) 处理嵌套对象的能力

不是所有的工具都能够识别嵌套对象，目前 EPUB 和 Open Document Format 都使用 ZIP 作为容器，能否识别嵌套对象是评价识别工具非常必要的一个指标。实验证明，DROID 能识别内嵌在 ZIP 中的所有文件，并且将其识别信息包含在识别结果集中。

(5) 处理复合对象的能力

能否识别复合对象也是评价格式识别工具非常重要的一个指标。测试证明，DROID 能识别 EPUB 和 Open Document Format、HTML 复合对象。

(6) 计算性能

DROID 在计算速度、内存和系统资源使用上满足基本的要求，主要从单文件识别和批量文件识别进行实验计算性能对比，如表 2 和表 3 所示：

表 2 单文件识别使用时间对比

文件数	单个文件最小执行时间(秒)	单个文件平均执行时间(秒)	单个文件最大执行时间(秒)	总耗时(秒)
11 892	11.19	11.90	23.02	141 514

表 3 批量文件识别使用时间对比

文件数	识别耗时(秒)	生成 CSV 耗时(秒)	总耗时(秒)
11 892	193	22	215

实验表明，DROID 在批量格式识别时的计算性能相当不错，能满足大数据量的文件格式识别需求。

5 结 语

总体来说，DROID 是一个优秀的开源工具，在应用中，仍需要更深入分析其执行原理和源码，有效利用它的工作机制和方式，这样才会在集成过程更高效、健壮地使用 DROID 进行文件格式识别，也能使 DROID 的优势得到更大的发挥。

(1) 大批量数据格式识别应选用DROID批量格式识别方式

DROID 有单文件识别和批量文件识别两种方式，从 DROID 的计算性能分析可见，在大批量数据对象识别中 DROID 工具以批量处理为优势。

(2) 注意 DROID 不同版本之间的差别

在实验中还发现，使用 DROID 的 V45 版的 SignatureFile 无法正确地完成文件格式的批量识别，将其升级到 V72 以上版本，才可完成文件格式的批量处理，因此在应用过程中不仅要考虑软件的版本还要注意 SignatureFile 版本之间的差异，使用过程中要根据实际需求考虑选用合适的版本进行格式识别。

(3) 充分考虑系统集成和更新

DROID 是一种开源工具，开源社区也在不断进行维护和版本升级。在实际应用中，笔者是把 DROID 作为一个模块集成到保存系统的存档流程中，从集成模块的后续维护和升级角度考虑，在集成方式上采用整体集成，以确保集成模块随着开源社区不断升级可以一起进行无缝升级。

参考文献：

[1] Abrams S. File Formats[OL].[2014-07-15]. http://www.dcc.

ac.uk/resources/curation-reference-manual/completed-chapters/file-formats.

[2] JHOVE - JSTOR/Harvard Object Validation Environment [EB/OL]. [2014-07-15]. http://jhove.sourceforge.net/.

[3] Chapter 4: DAITSS Preservation Services [EB/OL]. (2011-10-25). [2014-07-15]. https://share.fcla.edu/FDAPublic/DAITSS/Chapter_4_Preservation_Services.pdf.

[4] Portico Content Type Action Plan: Technical Artifacts [EB/OL]. (2009-08-05). [2014-07-15]. http://www.portico.org/digital-preservation/wp-content/uploads/2011/03/Portico-Content-Type-Action-Plan-Technical-Artifacts.pdf.

[5] Universal Archiving Module [EB/OL]. [2014-07-15]. http://rahvusarhiiv.ra.ee/en/universal-archives-module-2/.

[6] Kopal Library for Retrieval and Ingest[EB/OL].(2009-05-06). [2014-07-15]. http://kopal.langzeitarchivierung.de/index_koLibRI.php.de.

[7] File Profiling Tool (DROID) [EB/OL]. [2014-07-15]. http://www.nationalarchives.gov.uk/information-management/manage-information/policy-process/digital-continuity/file-profiling-tool-droid/.

[8] Integrating Planets and Fedora Commons [EB/OL]. (2010-08-11). [2014-07-15]. http://www.planets-project.eu/ publications/.

[9] Hitchcock S, Hey J, Brody T, et al.Laying the Foundations for Repository Preservation Services [R/OL]. (2007-03-07). [2014-07-15]. http://www.portico.org/digital-preservation/wp-content/uploads/2011/03/Portico-Content-Type-Action-Plan-Technical-Artifacts.pdf.

[10] PRONOM – The Online Registry of Technical Information [EB/OL]. [2014-07-15]. http://www.nationalarchives.gov.uk/pronom/.

[11] DROID: How to Use It and How to Interpret Your Results [OL]. [2014-08-18]. http://www.nationalarchives.gov.uk/documents/information-management/droid-how-to-use-it-and-interpret-results.pdf.

作者贡献声明：

王玉菊：设计研发方案，源码分析，二次开发，论文撰写，论文修改；

吴振新：提出研究命题和优化方案，论文最终版本修订；

孔贝贝：性能优化；

付鸿鹄：参与源码分析和系统开发。

收稿日期: 2014-07-21

收修改稿日期: 2014-09-22

Application of DROID About Format Identification in Long-term Preservation System

Wang Yuju　Wu Zhenxin　Kong Beibei　Fu Honghu

(National Science Library, Chinese Academy of Sciences, Beijing 100190, China)

Abstract: [**Objective**] Integrate open source file-format identification tool into Digital Preservation System (DPS) to get complex object format information. [**Context**] Based on the existing open source tools, to meet the practical requirements, the DPS needs choose appropriate tools for application integration. [**Methods**] Analyze and compare several open source file-format identification tools. According to the practical requirements, DROID has been chosen for the DPS. At the same time to meet the efficiency requirements of DPS, an idea of choosing DROID batch format identification of complex objects is proposed. [**Results**] Batch format processing module which is integrated with DROID is utilized to complete format identification of complex objects and technical metadata extraction. [**Conclusions**] DROID is an excellent open source tool, of which the automatic batch processing can meet the requirements of DPS.

Keywords: Format identification　Long-term preservation　Complex object